VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

NÁSTROJ PRO AUTOMATICKOU TVORBU PREZENTACÍ

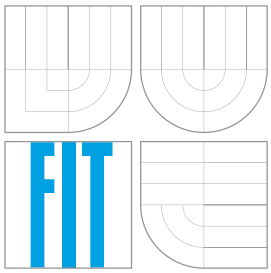
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MAROŠ UHERČÍK

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

NÁSTROJ PRO AUTOMATICKOU TVORBU PREZENTACÍ

TOOLS FOR AUTOMATIC CREATION OF PRESENTATIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MAROŠ UHERČÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. FILIP KADLČEK
SUPERVISOR

BRNO 2012

Abstrakt

Bakalářská práce je zaměřená na návrh a tvorbu nástroje pro automatickou tvorbu prezentací ve formátu OOXML. Práce obsahuje teoretický rozbor formátu OOXML se zaměřením na prezentace, analýzu a návrh nástroje a popis implementace. Praktickou částí práce je implementace navrženého nástroje. V závěru práce jsou uvedeny dosažené výsledky, demonstrace použití nástroje a návrh na pokračování projektu.

Abstract

This bachelor thesis deals with design and implementation of tool for automatic presentation creation in OOXML format. Work contains theoretic analysis of the OOXML format, focusing on presentation, analysis and design of tool and implementation description. Practical part of this work is implementation of designed tool. The final part shows obtained results, demonstration of tool usage and proposal of future works.

Klíčová slova

Office Open XML, OOXML, XML, PPTX, prezentace

Keywords

Office Open XML, OOXML, XML, PPTX, presentation

Citace

Maroš Uherčík: Nástroj pro automatickou tvorbu
prezentací, bakalářská práce, Brno, FIT VUT v Brně, 2012

Nástroj pro automatickou tvorbu prezentací

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Filipa Kadlčeka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Maroš Uherčík
16. mája 2012

Poděkování

Rád bych se poděkoval vedoucímu této práce Ing. Filipovi Kadlčekovi, který mi počas konzultací poskytoval své odborné rady a připomínky, čím přispěl k vypracování této práce.

Obsah

Kapitola 1

Úvod

Slovu prezentácia sa dá rozumieť vo viacerých významoch - výklad, ukážka, predstavenie, prezentačný program. V dnešnej dobe sa pod týmto pojmom často rozumie prezentácia v elektronickej podobe. Prezentácie v elektronickej podobe sa využívajú na prezentáciu výrobkov alebo služieb, ako podklad k prednáškam, prezentácií prác, zhrnutie zásad a postupov a podobne. Vytváranie prezentácií v elektronickej forme patrí medzi najčastejšie činnosti bežných užívateľov počítačov.

V dnešnej dobe najviac užívateľov využíva operačný systém Microsoft Windows, kde je najčastejšie používaným programom na tvorbu prezentácií Microsoft PowerPoint, ktorý je súčasťou kancelárskeho balíku Microsoft Office. Tento program používa na uloženie prezentácií štandard Office Open XML (ďalej OOXML)[8]. V programe typu Microsoft PowerPoint sa prezentácie vytvárajú interaktívne - užívateľ priamo vpisuje obsah do snímku (slide).

Užívatelia však často potrebujú vytvárať prezentácie denne na základe výsledkov svojej práce. Tu môže byť vytváranie prezentácie v programe typu Microsoft PowerPoint zdĺhavé a stereotypné vzhľadom k rovnakému spracovaniu rôznych údajov. Nástroj na automatickú tvorbu prezentácií by mohol pomôcť užívateľom pri vytváraní prezentácií z denných výsledkov práce. Takýto nástroj však musí podporovať nejaký štandardný formát, aby mohli byť prezentácie použiteľné. V práci sa teda zameriame na prezentácie, ktoré využívajú formát OOXML, ktorý je dnes najpoužívanejším formátom na uloženie elektronickej prezentácie. Ďalšou výhodou nástoja vzhľadom k iným programom na tvorbu prezentácií by mala byť nezávislosť na platforme, taktiež by nebola potrebná inštalácia veľkého balíku kancelárskych nástrojov ako pri Microsoft Office.

Práca sa bude zaoberať tvorbou prezentačného programu. Prezentačné programy sa rozdeľujú podľa toho, k akému účelu slúžia. Je možné ich rozdeliť na tri druhy:

- programy umožňujúce vytváranie prezentácií
- programy umožňujúce predvádzanie prezentácií
- kombinované (umožňujúce obe funkcie)

Táto práca bude zameraná na prvú skupinu prezentačných programov, a teda na vytváranie prezentácií. Hlavným cieľom práce je vytvoriť nástroj pre automatickú tvorbu prezentácií s jednoduchým rozhraním. Prezentácie vytvorené nástrojom musia spĺňať štandard OOXML. Nástroj musí byť nezávislý na platforme.

Kapitola 2 sa bude zaoberať formátmi súborov využívaných na tvorbu elektronických prezentácií. V kapitole 3 bude podrobne rozobraná štruktúra prezentačných súborov

v OOXML. Kapitola 4 sa bude zaoberať analýzou problémov, ktoré bude treba riešiť pri tvorbe nástroja a návrhom riešenia. V kapitole 5 sa budeme zaoberať konkrétnou implementáciou riešenia daného problému, vytvoreným rozhraním pre tvorbu prezentácií. V kapitole 6 sú zhrnuté výsledky, ktoré sme dosiahli, príklady vytváranie prezentácií v implementovanom nástroji. V kapitole 7 sa nachádza celkové zhrnutie práce a jej výsledkov. Taktiež je tu uvedený návrh na možnosti pokračovania projektu.

Kapitola 2

Formáty súborov

Táto kapitola sa bude zaoberať formátom prezentácií, ktoré budeme vytvárať. V kapitole je popísaný jazyk XML, ktorý je základom formátu OOXML a samotný formát OOXML.

2.1 Extensible Markup Language (XML)

XML je obecný značkovací jazyk. Značky jazyka umožňujú zachytiť v dokumente dôležité informácie o jeho štruktúre a význame. Najväčší prínos XML spočíva v tom, že v dokumentoch môžeme používať vlastné značky (tagy). Pre lepšie pochopenie jazyka bude uvedená stručná história jazyka, jeho vlastnosti, základy syntaxe a spôsoby spracovania [17].

História XML

Prvým známym značkovacím jazykom bol GML (Generalized Markup Language), ktorý vytvorili Charles Goldfarb, Edward Mosher a Raymond Lorie [7]. Museli sa vysporiadať s nekompatibilitou jednotlivých systémov a programov. Najľahšia cesta viedla práve cez vytvorenie nejakého obecného značkovacieho jazyka. Princíp GML sa osvedčil a v 80. rokoch na jeho základe začala vyvíjať štandardizačná organizácia ANSI jazyk, ktorý umožňoval definíciu vlastných značkovacích jazykov - užívateľ si mohol podľa potreby vytvoriť vlastnú sadu značiek, vhodnú pre daný druh dokumentu. Združenie GCA (Graphics Communications Association) už skôr vytvorilo štandardný formátovací jazyk GenCode. Mnohé ciele oboch projektov boli podobné, a preto sa obe aktivity spojili. Výsledkom bol jazyk SGML(Standard Generalized Markup Language), ktorý je definovaný v ISO norme 8879 z roku 1986 [7]. Jazyk SGML umožňuje definíciu vlastných značkovacích jazykov pomocou tzv. definíc typu(DTD). Asi najznámejšou aplikáciou SGML je HTML, ktorý sa využíva pre tvorbu webových stránok. Komplexnosť štandardu avšak zbrzdila jeho využitie. Behom desiatich rokov používania sa ukázalo, že sa v praxi používa len časť jeho možností. Táto najdôležitejšia podmnožina bola vybraná ako nový jazyk - XML. Jazyk XML si zachováva možnosť definovanie vlastných DTD, a teda i vlastných značiek pre jednotlivé skupiny dokumentov. Syntax zápisu dokumentu v XML je oproti SGML pomerne prísna, čo umožňuje omnoho ľahší a lacnejší vývoj aplikácií, ktoré umožňujú s týmto jazykom pracovať. Ako je zrejmé z histórie XML, tento jazyk pochádza z oblasti na uchovanie a spracovanie textových dokumentov. Pre tieto účely sa hodí výborne. Značky umožňujú zachytiť v dokumente dôležité informácie o jeho štruktúre a význame. XML je otvorený, flexibilný a preto veľmi vhodný na prácu s informáciami [12].

Vlastnosti XML

Bohaté využitie XML v dnešnej dobe je podmienené vlastnosťami tohto jazyka. Informácie o vlastnostiach jazyka sú získané zo zdrojov [12] a [9]. Základnými vlastnosťami jazyka sú:

- Štandardný formát pre výmenu informácií - v dnešnom globálnom svete nie je možné pre komunikáciu a výmenu informácií používať proprietárne formáty, ktoré sú viazané na konkrétny software. Je potrebné používať otvorený formát, ktorý nie je úzko zviazaný s konkrétnou platformou alebo technológiou. Takýmto formátom je XML.
- Medzinárodná podpora - jazyk dbá na potreby aj iných jazykov ako angličtiny. Ako znaková sada sa používa ISO 10646 - dokáže zahrnúť všetky znaky dnes používaných jazykov [10].
- Vysoký informačný obsah - pomocou značiek vyznačujeme v dokumente význam jednotlivých častí textu. Táto vlastnosť má veľký význam pre vyhľadávanie.
- Jednoduchá konverzia do iných formátov - k tejto konverzií slúžia tzv. štýly - súbor pravidiel alebo príkazov, ktoré definujú, ako sa dokument prevedie do iného formátu.
- Automatická kontrola štruktúry dokumentov - program ktorý kontroluje správnosť XML sa nazýva parser.
- Hypertext a odkazy - jazyk umožňuje vytváranie odkazov v rámci jedného dokumentu aj medzi dokumentmi navzájom. Ponúka bohaté možnosti tvorenia odkazov. Tvorba odkazov je popísaná v štandardoch XLink, XPointer a XPath.

2.1.1 Syntax XML

V tejto časti bude popísaná syntax jazyka XML. Nachádza sa tu základný popis syntaxe, použitie znakových sád a kódovania, popis definície typu dokumentu (DTD) a používanie menných priestorov. Tieto znalosti sú nevyhnutné pre tvorbu XML súborov.

Základ XML

Jazyk XML je case-sensitive (rozlišuje veľké a malé písmená). Princíp syntaxe je veľmi jednoduchý. Každý dokument sa skladá z elementov. Elementy sú základným stavebným kameňom každého dokumentu. Elementy sú do seba navzájom vnorené. V texte sa vyznačujú pomocou tzv. *tagov*. Väčšina elementov sa skladá z dvoch tagov - počiatočný a ukončovací. Názvy tagov zapisujeme medzi „<“ a „>“, pričom ukončovací tag má pred názvom znak „/“. Ak chceme použiť element, ktorý nebude mať žiadny obsah, môžeme využiť skráteného zápisu (napr. <`br /`>). Pri každom počiatočnom tagu môžu byť použité ešte atribúty. Atribúty sa používajú k spresneniu významu elementu. V XML je možné písať komentáre, ktoré sú označené značkami „<! − − komentár− − >“. Každý dokument musí byť celý obsiahnutý v jednom elemente. Na začiatku XML sa môže nachádzať deklarácia. V deklarácií sa uvádza verzia XML a použité kódovanie.

Znakové sady a kódovanie

Znaková sada definuje, aké znaky a pod akým číslom máme k dispozícii. Jednou z najstarších a najznámejších znakových sád je ASCII. Táto znaková sada je 7bitová a obsahuje znaky s kódmi 0 až 127. Potrebám angličtiny ASCII vyhovovalo. Pre ostatné jazyky tu

však chýbali niektoré písmená a znaky. Vznikli preto rozsiahlejšie znakové sady. XML používa znakovú sadu ISO 10646 (32 bitová sada), pretože dokáže zahrnúť všetky znaky dnes používaných jazykov [10]. Pretože pre väčšinu dokumentov by bolo priame použitie ISO 10646 zbytočným plytvaním - jeden znak by bol na štyri bajty - štandardne sa v XML používa kódovanie UTF-8. Toto kódovanie je identické s ASCII. Ďalšie znaky nad rámec ASCII sú kódované do sekvencií 2 až 6 bajtov. Pokiaľ chceme v dokumente používať kódovanie, musíme ho špecifikovať pomocou XML deklarácie, ktorá je uvedená v prvom riadku dokumentu. Použité kódovanie určíme pomocou parametru *encoding*.

Definícia typu dokumentu (DTD)

Ako už bolo povedané, XML dokument umožňuje definíciu typu dokumentu. Táto definícia hovorí, aké elementy a atribúty je možné v dokumente použiť. DTD nám umožňuje kontrolovať správnosť štruktúry dokumentov. Použitie DTD má hneď dve výhody. Pomocou parseru môžeme kontrolovať, či má dokument štruktúru odpovedajúcu DTD. Ďalšia výhoda sa uplatní pri použití štandardných DTD, pre ktoré máme k dispozícii množstvo jednoúčelových nástrojov. Ak chceme pre dokument DTD určiť, musíme použiť deklaráciu typu dokumentu (DOCTYPE).

Menné priestory (XML namespaces)

Pomocou menných priestorov je možné v XML dokumente kombinovanie viacero sád značiek. Každá sada značiek je jednoznačne definovaná svojou URI adresou. Ak chceme v nejakom elemente a jeho podelementoch používať elementy patriace do určitej sady značiek, musíme to špecifikovať pomocou špeciálneho atribútu `xmlns`.

Na záver tejto časti si uvedieme príklad syntakticky správneho dokumentu s použitím vyššie zmienených zápisov. V Tabuľka 2.1 na prvom riadku sa nachádza deklarácia, kde je uvedená verzia a kódovanie dokumentu. Element `p:sld` demonštruje použitie menného priestoru. Element `p:cNvPr` je uvedený pomocou skráteného zápisu a využitia atribútov. Následne element `p:cNvGrpSpPr` obsahuje konkrétny text.

```
<?xml version="1.0" encoding="utf-8"?>
<p:sld xmlns:a="http://schemas.openxmlformats.org/drawingml/2006/main" />
  <p:cSld>
    <p:spTree>
      <p:nvGrpSpPr>
        <p:cNvPr id="1" name="meno"/>
        <p:cNvGrpSpPr>Obsah elementu</p:cNvGrpSpPr>
      </p:nvGrpSpPr>
    </p:spTree>
  </p:cSld>
<p:sld>
```

Tabuľka 2.1: XML dokument

2.1.2 Spracovanie XML

Pre prácu s XML súbormi existuje množstvo nástrojov. Veľmi často sú využívané parsery XML. Parsery sa delia na parsery typu DOM a SAX. Ďalšou možnosťou je dotazovací jazyk XPath. Tieto tri prístupy budú popísané podrobnejšie:

- SAX(Simple API for XML) - umožňuje sériový prístup k XML. Ide o tzv. prúdové spracovanie, pri ktorom sa dokument rozdelí na jednotlivé jeho časti (počiatočné a koncové značky, obsahy elementov, komentáre atď.). Postupne sa potom volajú jednotlivé udalosti, ktoré ohlasujú nájdenie konkrétnej časti. Spôsob spracovania týchto jednotlivých udalostí je potom úplne v kompetencii programátora [6].
- DOM(Document Object Model) - je objektovo orientovaná reprezentácia XML alebo HTML dokumentu. Je to API (Application Programming Interface), ktoré umožňuje prístup alebo modifikáciu obsahu, štruktúry alebo štýlu dokumentu alebo jeho častí. DOM umožňuje prístup k dokumentu ako ku stromu, čo je zároveň štruktúra používaná vo väčšine XML parserov. Táto technológia, nazývaná *grove* (Graph Representation Of property Values), vyžaduje nahradenie celého parsovaného dokumentu do pamäti. Optimálne použitie je tam, kde sa k jednotlivým elementom dokumentu sa pristupuje v náhodnom poradí alebo opakovane [16].
- XPath - je to jazyk, pomocou ktorého sa dá adresovať časti XML dokumentu. Pomocou tohto jazyka je možné z XML dokumentu vyberať jednotlivé elementy a pracovať s ich hodnotami a atribútmi. XPath modeluje XML dokument ako stromovú štruktúru, kde jednotlivé uzly sú rôznych typov, ako napríklad elementy, atribúty alebo text. Základnou syntaktickou konštrukciou v XPath je výraz, ktorého vyhodnotením vzniká objekt. Základnými typmi objektov sú množina uzlov, logická hodnota, číslo a reťazec. XPath je možné použiť k získavaniu uzlov s určitými vlastnosťami a k testovaniu, či určité uzly majú nejakú vlastnosť. Vyhodnotenie výrazu prebieha v určitom kontexte. Kontext pozostáva z kontextového uzlu, pozície a veľkosti kontextu, množiny naviazaných premenných, knižnice funkcií a množiny platných deklarácií menných priestorov. Kontext je určovaný štandardom, ktorý XPath využíva [18].

2.2 Office Open XML (OOXML)

OOXML je formát pre ukladanie dát založený na technológií XML a kompresii ZIP. Formát bol vyvinutý spoločnosťou Microsoft. Slúži na ukladanie súborov kancelárskych dokumentov. OOXML využívajú aplikácie zo sady kancelárskych balíkov Microsoft Office(napr. Microsoft Word, Microsoft Excel, Microsoft PowerPoint). Jeho predchodcom boli proprietárne binárne formáty (.doc, .ppt, .xls). V OOXML sa zmenili prípony súborov pripojením písmena "x"(.docx, .pptx, .xlsx). Súbor vo formáte OOXML je teda komprimovaný balíček, ktorý obsahuje množstvo XML súborov, taktiež obrázky a súbory do dokumentu vložené. Prechodom na formát OOXML sa zmenšila veľkosť súborov, zjednodušilo sa riešenie problémov so súbormi, uľahčila sa ochrana citlivých informácií v dokumentoch. Informácie o štandarde sú získané zo zdroja [11].

Štandardizácia OOXML

Formát OOXML bol štandardizovaný v decembri roku 2006 organizáciou ECMA (ECMA-376) [8]. Tento dokument špecifikuje reprezentáciu dokumentov vo formáte

OOXML (XML schémy, výrazy pre textové, prezentačné a tabuľkové dokumenty). Štandard obsahuje päť častí, ktoré obsahujú 6546 strán. Jedná sa teda o veľmi rozsiahlu špecifikáciu. V roku 2008 bol formát OOXML štandardizovaný organizáciami ISO a IEC ako ISO/IEC 29500[11]. Behom štandardizácie došlo k niekoľko podstatným zmenám oproti ECMA-376 na základe požiadavky národných štandardizačných inštitútov. Najdôležitejšou zmenou bolo definovanie dvoch typov dokumentov - Strict a Transitional. Vo verzii Strict sa nenachádzajú elementy, ktoré sú zaťažené spätnou kompatibilitou smerom k starším dokumentovým formátom. Verzia Transitional obsahuje aj elementy, ktoré sú nevyhnutné pre spätnú kompatibilitu.

Výhody OOXML:

- Formát nezávislý na platforme a aplikácií - pre vytvorenie a modifikáciu súborov nie je potrebné mať zakúpený špecializovaný software.
- Otvorenosť, žiadne licenčné poplatky - špecifikácie pre formáty sú zverejnené. Je umožnený prístup a tvorenie súborov bez licenčných poplatkov.
- Interoperabilita (schopnosť rôznych systémov vzájomne spolupracovať) - použitý formát XML je jednoduchý na manipuláciu v bežných programovacích jazykoch. Preto nie je náročná manipulácia, prípadné vytváranie súborov v OOXML.
- Robustnosť.
- Efektivita - zmenšenie veľkosti súborov, zrýchlenie vyhľadávania v texte.
- Bezpečnosť.
- Spätná kompatibilita.

Typy XML schém

XML schémy formátu OOXML sa rozlišujú podľa typu dokumentu, ktorý ukladajú. V XML súboroch potom pomocou menných priestorov (namespaces) môžeme zistiť aká schéma bola použitá. Využívajú sa štyri základné schémy:

- `WordprocessingML` - používa sa na vytváranie textových dokumentov (koreňový element <book>). Súbory s príponou .docx.
- `SpreadsheetML` - používa sa na vytváranie tabuľkových dokumenty (koreňový element <workbook>). Súbory s príponou .xlsx.
- `PresentationML` - jazyk využívaný na vytváranie prezentácií (koreňový element <presentation>). Týmto jazykom sa podrobnejšie budeme zaoberať v nasledujúcej kapitole.
- `DrawingML` - využívajú ho vyššie zmienené typy pre špecifikáciu lokácie a vzhľadu daného dokumentu.

Kapitola 3

PresentationML a jeho štruktúra

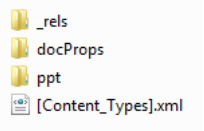
Táto kapitola je zameraná na bližšie oboznámenie sa s PresentationML, a teda balíkom využívaným pri ukladaní prezentácií. Súbor prezentácie vytvorenej v štandarde OOXML má príponu súboru `.pptx`. Z predošlej kapitoly je známe, že súbory v OOXML sú balíčkami ZIP, ktoré obsahuje XML súbory a súbory potrebné na zobrazenie všetkých objektov v dokumente. Pre nahliadnutie do štruktúry súboru stačí rozbaliť súbor s príponu `.pptx`. Po rozbalení tohto balíčku máme možnosť prehliadnuť si vnútornú štruktúru súboru. V tejto kapitole budú rozobraté podrobné informácie o PresentationML. Informácie sú získané zo zdrojov [15], [11].

3.1 Štruktúra a popis súborov

V tejto časti je popísaná štruktúra a jednotlivé časti prezentácií vo formáte OOXML. Nachádza sa tu detailný popis najdôležitejších elementov PresentationML a DrawingML, ktoré sa pri vytváraní prezentácií používajú. Informácie v tejto časti sú nevyhnutné pre vytváranie prezentácií vo formáte OOXML. Po prečítaní čitateľ bude oboznámený s usporiadaním štruktúry súborov v prezentácii a hlavným obsahom jednotlivých XML súborov.

3.1.1 Obsah balíku prezentácie

Po rozbalení prezentácie môžeme vidieť základnú štruktúru, ktorá je znázornená na Obr. 3.1. Nachádzajú sa tu tri zložky a jeden XML súbor. V tejto časti budú popísané zložky `_rels`, `docProps` a súbor `[Content_Types].xml`. Obsahu zložky `ppt` bude venovaná samostatná časť.



Obr. 3.1: Základná štruktúra prezentácie

Súbor Content_Types.xml Súbor obsahuje dva typy definície pre obsah balíčku ZIP:

- Typy súborov nachádzajúcich sa v balíčku ZIP (XML, RELS) a ďalších (napr. grafický súbor PNG uvedený na obrázku) - označované ako *Default Extension*.
- Kľúčové typy obsahu (označované ako *Override PartName*). Tento typ je vyjadrený ako cesta k časti XML v balíčku ZIP (napr. `/ppt/presentation.xml`).

Označenia *Default Extension* a *Override PartName* tvoria definíciu typu obsahu [14].

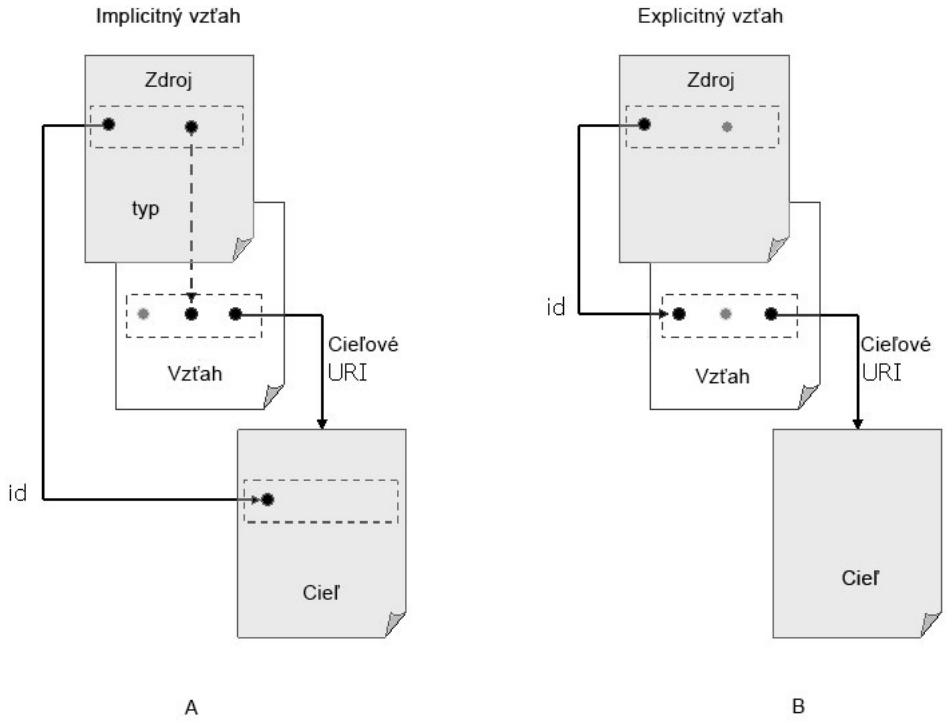
Súbory pre vzťahy Pre každú úroveň štruktúry balíčku sa môžu vyskytovať informácie o vzťahoch medzi jednotlivými časťami balíčka. Informácie o vzťahoch sa vyskytujú v prípade, ak sú časti balíčku na sebe závislé. Nato aby bol vzťah jednoznačne určený musí obsahovať atribúty `id`(identifikácia), `Type` (typ vzťahu) a `Target` (cieľový súbor). Príklad súboru určujúceho vzťahy je zobrazený v Tabuľka 3.1. V koreňovej zložke sa nachádza podzložka `_rels`, ktorá obsahuje súbor `.rels`. V tomto xml súbore sa nachádzajú informácie, pomocou ktorých aplikácia zistí, kde vyhľadať súčasti prezentácie. Vzťahy v tejto najvyššej úrovni obsahujú odkazy na súbory `/ppt/presentation.xml`, `docProps/thumbnail.jpeg`, `docProps/app.xml`, `docProps/core.xml`, ktoré budú popísané v nasledujúcej časti textu. V podzložke `ppt` sa taktiež nachádza zložka `_rels`. V zložke nájdeme súbor `presentation.xml.rels`, ktorý reprezentuje vzťahy súboru `presentation.xml` so súbormi typu slide, SlideMaster, SlideLayout, Theme, presProps, viewProps, tableStyles atď. V jednotlivých podzložkách môžeme opäť nájsť podzložky obsahujúce súbory so vzťahmi. Typicky sú to vzťahy pre súbory typu Slide, SlideMaster, SlideLayout, Theme. Vzťahy, ktoré môžu vznikať medzi jednotlivými časťami dokumentov sú znázornené v Tabuľka 3.2. V prezentácií existujú dva typy vzťahov:

- Explicitné - identifikátor v XML dokumente odkazuje na vzťah, ktorý odkazuje priamo na požadovaný objekt.
- Implicitné - identifikátor v XML dokumente je použitý pre lokáciu daného elementu vo vnútri cieľového dokumentu.

Rozdiel medzi vzťahmi je znázornený na Obrázok 3.2.

```
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/
     relationships">
     <Relationship Id=”identifikacia” Type=”typ” Target=”odkaz” />
     .
     .
     .
</Relationships>
```

Tabuľka 3.1: Vztah



Obr. 3.2: Implicitný (A) a explicitný (B) vzťah

Podzložka docProps Táto podzložka obsahuje dva súbory, ktoré slúžia na ukladanie metadát o OOXML dokumente:

- `app.xml` - ukladá štatistiky o dokumente (počet slidov, počet poznámok, zdrojová aplikácia atď.).
- `core.xml` - ukladá všeobecné štatistiky (užívateľ, ktorý dokument vytvoril; časové údaje - modifikácia, vytvorenie atď.).

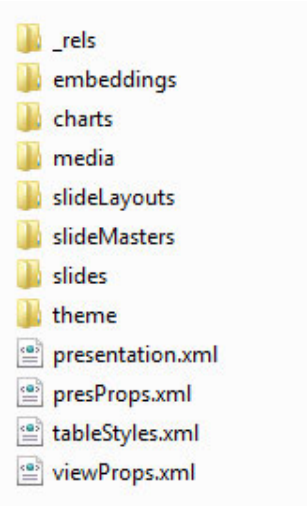
Taktiež sa tu môže nachádzať súbor `thumbnail.jpeg`. Tento súbor sa využíva ako obrázok pre ikonu súboru (na obrázku je úvodný slide prezentácie).

Part	Relationship Target of	Root Element
Comment Authors	Presentation	cmAuthorLst
Comments	Slide	cmLst
Handout Master	Presentation	handoutMaster
Notes Master	Notes Slide, Presentation	notesMaster
Notes Slide	Slide	notes
Presentation	PresentationML package	presentation
Presentation Properties	Presentation	presentationPr
Slide	Presentation	sld
Slide Layout	Slide Master, Notes Slide, Presentation, Slide, Slide Master	sldLayout
Slide Master	Presentation, Slide Layout	sldMaster
Slide Synchronization Data	Slide	sldSyncPr
User-Defined Tags	Presentation, Slide	tagLst
View Properties	Presentation	viewPr

Tabuľka 3.2: Tabuľka vzťahov jednotlivych častí

3.1.2 Zložka ppt

Typickým obsah zložky je zobrazený na Obr. 3.3. Nachádza sa tu niekoľko xml dokumentov a ďalšie zložky reprezentujúce obsah prezentácie, ktoré tu budú popísané.



Obr. 3.3: Typický obsah zložky `ppt`

Dokument presentation.xml Nachádzajú sa tu vlastnosti, ktoré sa týkajú celej prezentácie. Najpodstatnejšie vlastnosti (elementy xml) si popíšeme bližšie:

- `sldIdLst` - zoznam odkazov na vzťahy, ktoré spájajú prezentáciu s položkami reprezentujúcimi slidy.
- `sldMasterIdLst` - špecifikuje slideMaster, ktorý je použitý v prezentácii.

- `sldSz` - určuje veľkosť slidu a pomer strán (napr. screen4x3).
- `defaultTextStyle` - špecifikuje predvolený(default) štýl textu použitý v prezentácií. Tento štýl sa použije pokiaľ nie je predefinovaný v nastaveniach slidu. V tomto elemente môžeme definovať štýl textu podľa konkrétnych úrovní pomocou elementov `lvl1pPr-lvl9pPr`, ktoré však už patria do jazyka DrawingML.

Dokument viewProps.xml Nachádzajú sa tu vlastnosti, ktoré majú vplyv na zobrazovanie prezentácie. Koreňovým elementom je `viewPr`. Typicky sa tu nachádza špecifikácia zobrazenia pre normálny mód zobrazenia - `normalViewPr` (Normal View Properties), zobrazenie poznámok `notesViewPr` (Notes View Properties) a zobrazenie pri prezentovaní `slideViewPr` (Slide View Properties).

Dokument presProps.xml V súbore sú definované vlastnosti prezentácie. Koreňovým elementom je `p:presentationPr`.

Dokument tableStyles.xml V súbore sú uložené informácie o štýle tabuliek použitých v prezentácií. Štýly definujú vlastnosti ako napríklad farbu riadkov a stĺpcov, hlavičky stĺpcov, štýly textov.

Zložka media Nachádzajú sa tu súbory, ktoré sú do prezentácie vložené. Jedná sa napríklad o obrázky, videá a podobne. Konkrétne slidy sú potom naviazané na tieto súbory.

Zložka slideMasters V zložke sa nachádza použité predlohy slidov (Slide Masters) prezentácie. V dokumente typu SlideMaster sa nachádzajú elementy, ktoré popisujú objekty v prezentácií a ich formátovanie. Nájdeme tu odkazy na rozloženia slidov (Slide Layouts) v elemente `sldLayoutIdLst`. V SlideMaster nájdeme dva hlavné elementy:

- `cSld` - špecifikuje prvky slidov (napr. geometrické elementy, textové polia).
- `txStyles` - špecifikuje formátovanie textu v objektoch.

Zložka slideLayouts Nachádzajú sa tu XML súbory pre použite slideLayouty (rozložnie slidov) v prezentácií. V dokumente typu SlideLayout sa nachádza špecifikácia výzoru konkrétneho slidu. SlideLayout sa môže viazať na ktorýkoľvek existujúci slide. Všetky elementy použité v slide by mali byť mapované na slideLayout. Obsah slideLayout a slide môže byť takmer zhodný. Rozdielom je, že v slide nájdeme konkrétne údaje a v slideLayout návrh, do ktorého sa údaje mapujú.

Zložka slide Nachádzajú sa tu XML súbory slidov v prezentácii. V dokumente typu Slide sa nachádza konkrétny obsah slidu. Ako už bolo spomenuté obsah by mal byť mapovaný na rozloženie slidu (Slide Layout). Koreňový element sa nazýva `p:sld`. Najčastejšie používanými elementmi sú:

- `p:cSld` (Common Slide Data) - do elementu sa špecifikujú informácie o slide. Tento prvok sa využíva vo všetkých typoch slidov. V prvku sa využívajú vlastnosti, ktoré sú nezávislé na type slidu.
- `p:spTree` (Shape Tree) - v tomto elemente sa nachádzajú všetky geometrické (tvarové) elementy. Do týchto elementov sa vkladajú texty, obrázky atď.
- `p:sp` (Shape) - je podelementom `p:spTree`. Je všeobecným geometrickým elementom, do ktorého je možné vkladať text.

- `p:nvSpPr` (Non-Visual Properties for a Shape) - tento element definuje neviditeľné vlastnosti elementu `p:sp`.
- `p:cNvPr` (Non-Visual Drawing Properties) - tento element definuje neviditeľné vlastnosti elementov patriacich do DrawingML.
- `p:spPr` (Shape Properties) - definuje vlastnosti elementu `p:sp`.
- `p:cNvSpPr` (Non-Visual Drawing Properties for a Shape) - tieto vlastnosti využíva zobrazujúca aplikácia pre určenie ako sa má geometricky objekt zobraziť.
- `p:txBody` (Shape Text Body) - element definuje existenciu textu, ktorý sa má vyskytovať v konkrétnom `sp` elemente.
- `p:pic` (Picture) - je podelementom `p:spTree`. Využíva sa na vkladanie obrázkov do prezentácie. Pre obrázky je taktiež možné definovanie vlastností pomocou `nvPicPr`, `nvPr` a `cNvPicPr`. Tieto elementy majú obdobný význam ako pri prvku `sp`.
- `p:graphicFrame` (Graphic Frame) - je podelementom `spTree`. Využíva sa na vkladanie rôznych objektov ako grafy, tabuľky a pod.

Zložka embeddings Nachádzajú sa tu vložené dokumenty. Objekty z tejto zložky sú v slidoch vložené do grafických rámcov (element graficFrame). Následne element, ktorý špecifikuje údaje pre grafický rámec, musí obsahovať odpovedajúci odkaz URI, podľa obsahu súboru. Napríklad sa do tejto zložky ukladajú XLSX súbory (tabuľkové dokumenty), ktoré slúžia ako zdroj dát pre graf. Potom URI pre graficData je "http://schemas.openxmlformats.org/drawingml/2006/chart".

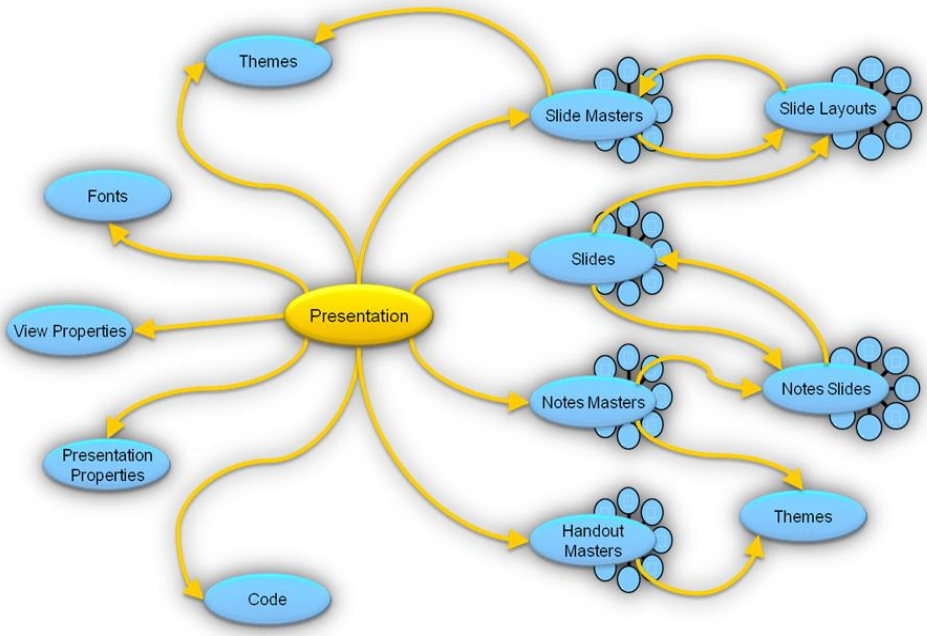
Zložka charts Nachádzajú sa tu XML súbory, ktoré reprezentujú grafy v prezentácií. Tieto súbory sa viažu na XLSX súbory uložené v zložke `embeddings`. Z XLSX súborov prezentácia čerpá údaje pre graf. Pri vkladaní grafu sa vytvárajú grafické rámce, ktoré majú typ dát určený ako graf. Pri grafoch je nutné určiť typ grafu - spojnicový, koláčový, stĺpcový atď. Pre bližšie zoznámenie s vkladaním grafov do prezentácie budú popísané najdôležitejšie elementy XML súborov pre grafy:

- `c:chartSpace` (Chart Space) - hlavný element, v ktorom sú uvedené príslušné menné prietory pre XML.
- `c:chart` (Chart) - v elemente je špecifikovaný celý graf.
- `c:legend` (Legend) - v elemente je špecifikovaná legenda grafu.
- `c:plotArea` (Plot Area) - v elemente sú špecifikované údaje pre graf a typ grafu.
- `c:barChart` (Bar Charts) - výskyt tohto elementu špecifikuje, že sa bude jednať o stĺpcový graf.
- `c:pieChart` (Pie Charts) - výskyt tohto elementu špecifikuje, že sa bude jednať o koláčový graf.
- `c:lineChart` (Line Charts) - výskyt tohto elementu špecifikuje, že sa bude jednať o spojnicový graf.
- `c:ser` (Chart Series) - element reprezentujúci sériu údajov (popis a údaje grafu).
- `c:tx` (Series Text) - v elemente sú špecifikované mená skupín údajov v grafe.
- `c:cat` (Category Axis Data) - v elemente sú špecifikované mená kategórií údajov v grafe.
- `c:val` (Values) - v elemente sú špecifikované konkrétne údaje pre graf.

3.2 Typická štruktúra prezentácie

Typická štruktúra prezentácia obsahuje jeden Slide Master a tému, niekoľko slidov, z ktorých každý má svoj Slide Layout a každý môže obsahovať komentáre a poznámky. Najpoužívanejšie elementy boli popísané v časti 3.1. Na Obr. 3.4 je zobrazená architektúra dokumentu prezentácie. Na obrázku môžeme vidieť z akých častí sa prezentácia skladá a v akých vzťahoch sa jednotlivé časti nachádzajú. Samotná prezentácia obsahuje svoju tému (Themes), definíciu fontov (Fonts), viditeľných (View Properties) a neviditeľných (Presentation Properties) vlastností. Ďalej prezentácia obsahuje:

- Predlohy slidov (Slide Masters) - viažu sa na rozloženie slidov (Slide Layouts).
- Slidy - viažu na svoje rozloženia (Slide Layouts) a na poznámky (Notes Slides).
- Predlohy pre poznámky (Notes Masters) - viažu sa na poznámky (Notes Slides) a témy (Themes).
- Predlohy podkladov (Handout Masters) - viažu sa na témy (Themes).



Obr. 3.4: Architektúra dokumentu prezentácie

Kapitola 4

Analýza a návrh

V tejto kapitole sa budeme zaoberať analýzou a návrhom nástroja pre automatickú tvorbu prezentácií. Bude tu popísaná analýza problému a návrh štruktúry nástroja. Na záver budú stanovené ciele, ktoré chceme dosiahnuť.

4.1 Analýza problému

V tejto časti sa zameriame na analýzu problémov, ktoré pri implementácií nástroja bude nutné riešiť. Ako už bolo spomenuté formát OOXML je založený na XML súboroch, ktoré sú vo vzájomných vzťahoch. Pri implementácií nástroja na tvorbu prezentácií v OOXML je teda nutné zvoliť vhodný programovací jazyk, ktorý by umožňoval dobrú prácu s XML.

4.1.1 Práca s XML

Podstatnú časť implementácie bude tvoriť práca s XML súbormi. Pri implementácií bude vhodné zvoliť objektovo orientovanú reprezentáciu XML dokumentov (DOM). Tento prístup uchováva v pamäti celú štruktúru dokumentu, čo by mohlo byť obmedzujúce pri práci s veľkými XML dokumentmi. V našom prípade však táto možnosť nehrozí. Veľkou výhodou objektového prístupu je možnosť modifikácie a pridávania elementov do dokumentov. Veľmi vhodné bude využívať knižnicu pre prácu s XML, ktorá podporuje dotazovací jazyk XPath. Pomocou tohto jazyka urýchlime vyhľadávanie v elementoch, editáciu a tvorbu nových elementov.

Druhou možnosťou je prístup k XML dokumentom pomocou parseru typu SAX. Jedná sa o prúdové spracovanie. Dokument sa nenahráva do pamäti. Spracovanie dokumentu prebieha postupne. Nevýhodou tohto prístupu je nemožnosť modifikácie a pridávania elementov v dokumentoch. Tento prístup je vhodný pri spracovaní veľkých súborov a pre získavanie informácií z dokumentov. Pre tvorbu, editáciu a mazanie elementov v dokumente je vhodnejšie zvoliť objektovo orientovanú reprezentáciu XML dokumentov (DOM), a teda bude využitá pri implementácií nástroja.

4.1.2 Problémy spojené s formátom OOXML

OOXML a jeho presentationML package má veľké množstvo objektov a vlastností, ktoré do prezentácie môžeme zaradiť. Úlohou v tejto časti bude vybrať najpoužívanejšie časti presentationML využívané v praxi a analyzovať podproblémy pri vytváraní prezentácií. Tvorbu prezentácie vo formáte OOXML je možné rozdeliť na niekoľko podproblémov:

- Obecné problémy:
  - Premena jednotiek vložených od užívateľa (pixely) na jednotky využívané v OOXML pri zadávaní rozmerov objektov.
  - Vytváranie identifikátorov a mien objektov na základe už existujúcich objektov.
  - Vytváranie vzťahov medzi jednotlivými časťami prezentácie.
- Vytvorenie šablóny prezentácie - väčšina objektov v prezentácií predstavuje XML súbor. Tvoriť všetky XML súbory nanovo pri každej novej prezentácii by bolo neefektívne a veľmi náročné. Pre XML súbory by bolo vhodné vytvoriť šablónu, do ktorej by sa pridávali elementy podľa vstupu užívateľa. Šablóna by predstavovala základnú štruktúru prezentácie. Šablóna by teda mala obsahovať každý element, ktorý nástroj podporuje. V šablóne by sa nachádzali predvolené nastavenia, čo by uľahčilo prácu užívateľa s nástrojom.
- Pridávanie slidu do prezentácie - pri pridávaní slidu do prezentácie je potrebné si uvedomiť, z akých úkonov sa táto činnosť skladá:
  - Vytvorenie samotného XML súboru pre slide - `/ppt/slides/slideXY.xml` (kde `XY` je poradové číslo slidu).
  - Vytvorenie súboru pre vzťahy slidu - `/ppt/slides/_rels/slideXY.xml.rels` - musí obsahovať minimálne odkaz na rozloženie slidu (slideLayout).
  - Pridanie referencie do elementu `p:sldIdLst` v súbore `/ppt/prezentation.xml`.
  - Pridanie referencie do súboru `[Content_types].xml`.
- Vloženie textu - pri vkladaní textu je potrebné si uvedomiť, že text sa bude vkladať do elementu `p:sp`, teda do textového poľa. Najskôr si musíme vytvoriť textové pole a potom do neho vložiť text.
- Vloženie grafu - graf v OOXML reprezentuje XML súbor s názvom `/ppt/charts/chartXY.xml`. Údaje do XML súboru je potrebné získať z XLSX súboru (tabuľkový dokument). Vstupom pri vkladaní grafu bude typ grafu, umiestnenie grafu, veľkosť grafu na slide, údaje z XLSX súboru, ktoré sa majú použiť (konkrétne bunky alebo rozsah buniek). Pri vkladaní grafu bude potrebné implementovať nasledujúce činnosti:
  - Vybratie údajov z XLSX súboru na základne zadaných buniek od užívateľa - toto zahŕňa rozbalenie XLSX súboru, nájdenie súboru, v ktorom sa údaje nachádzajú, vybratie hodnôt zo zadaných buniek. XLSX súbory pochádzajú z aplikácie Microsoft Excel. Tieto súbory sú uložené v OOXML za využitia SpreadsheetML a majú veľmi podobnú štruktúru ako prezentačné súbory. Po rozbalení súboru sa v zložke `xl` nachádza obsah tabuľkového dokumentu. V podzložke `worksheets` sa nachádzajú jednotlivé listy dokumentu (napr. `sheet1.xml`). V dokumente tohto typu sa nachádzajú elementy, ktoré ukladajú údaje bunky a v atribútoch ukladajú ich označenie.
  - Nakopírovanie XLSX súboru do zložky `/ppt/embeddings`.
  - Pridanie vzťahu slidu a grafu do súboru pre vzťahy slidu.
  - Vytvorenie samotného súboru `/ppt/charts/chartXY.xml` za použitia údajov z XLSX súboru.

  - Pridanie vzťahu grafu a XLSX súboru do `/ppt/charts/_rels/chartXY.xml.rels`.

- Vloženie obrázku - pri vkladaní obrázku bude užívateľ zadávať meno obrázku (cestu k nemu), veľkosť a umiestnenie na slide. Pre vloženie je potrebné implementovať nasledujúce činnosti:
  - Nakopírovanie obrázku do dočasného priečinku.
  - Zmenšenie alebo zväčšenie obrázku.
  - Nakopírovanie upraveného obrázku do zložky `/ppt/media/`.
  - Vloženie elementu `p:pic` do slidu a pridanie vzťahu slidu a obrázku do súboru `/ppt/slides/_rels/slideXY.xml.rels`.
- Vloženie čísla slidu - číslovanie slidu sa bude uplatňovať buď na celú prezentáciu alebo jednotlivé slidy.
- Vloženie zadanej grafickej témy - grafickú tému definujú súbory `/ppt/themes/themeXY.xml` a `/ppt/slideMasters/slideMasterXY.xml`. Pri výbere témy sa tieto súbory vyberú zo šablóny prezentácie. Taktiež k niektorým témam sa môžu vzťahovať ďalšie súbory, ako napríklad obrázky. Je potrebné ich nakopírovať do príslušnej zložky (zložka `media`) a pridať vzťah k týmto súborom do súboru `/ppt/themes/_rels/themeXY.xml.rels`.
- Vkladanie tabuliek - tabuľku je potrebné vložiť do grafického rámca (element `p:graphicFrame`). Pri vkladaní bude užívateľ zadávať veľkosť a pozíciu grafického rámca a dvojrozmerné pole, ktoré bude reprezentovať údaje tabuľky.
- Vytvorenie celej štruktúry súboru - vytvorenie zložiek a XML súborov tvoriacich štruktúru súboru.

4.2 Návrh štruktúry nástroja

V tejto časti sa budeme venovať návrhu štruktúry nástroja. Požiadavky pre vytváranie prezentácií naznačujú, že pri implementácií nástroja by bolo vhodné zvoliť objektovo orientovaný prístup. Tento prístup nám umožní reprezentovať objekty v prezentácií, ich vlastnosti a operácie, ktoré nad nimi môžeme vykonávať. Bude teda nutné reprezentovať nasledujúce objekty prezentácie:

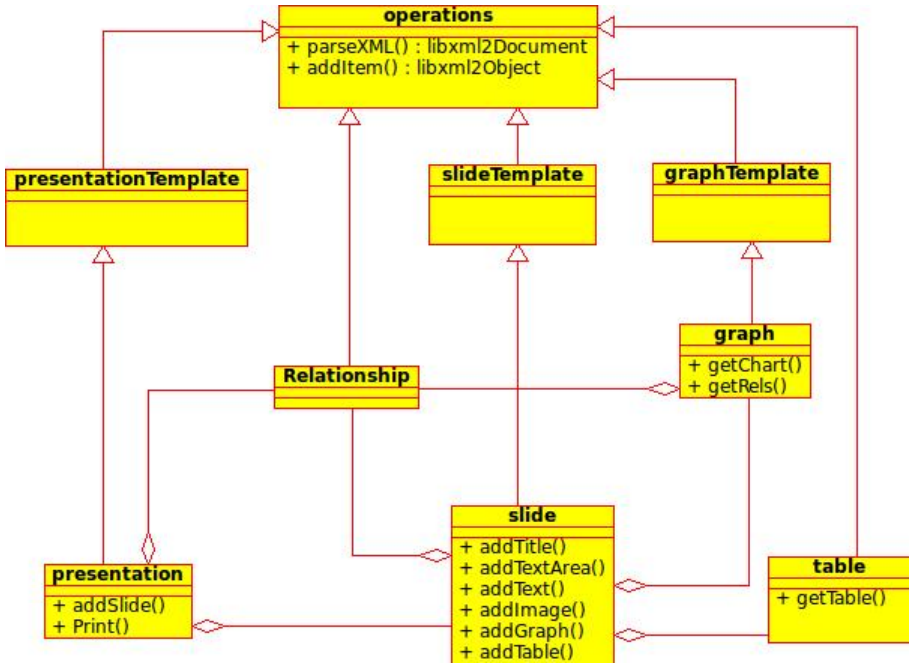
- Prezentácia - reprezentácia celej štruktúry prezentácie, zapuzdrenie všetkých objektov v prezentácii.
- Slide - reprezentácia jedného slidu, zapuzdrenie objektov vložených do slidu.
- Textové pole - reprezentácia textového poľa, jeho rozmerov a umiestnenia v slide.
- Obrázok - reprezentácia obrázku, jeho rozmerov a umiestnenia v slide.
- Graf - reprezentácia grafu, jeho typ, vstupné údaje, popisky dát.
- Tabuľka - reprezentácia tabuľky, jej údajov a a rozmerov.
- Obecný vzťah medzi elementmi - reprezentácia vzťahu medzi dvoma entitami.

Navrhnutá štruktúra je zobrazená na diagrame tried Obr. 4.1. Diagram nie je detailným popisom tried. Sú tu uvedené len podstatné metódy tried a vzťahy medzi triedami. Na obrázku je vidieť niekoľko najdôležitejších tried. Pre lepšie pochopenie diagramu tried, si stručne popíšeme jednotlivé triedy a vzťahy medzi triedami.

Trieda `operations` je trieda, od ktorej väčšina tried dedí implementáciu metód, ktoré budú nevyhnutné pre prácu s XML súbormi. Týmto sa vyhneme opätovnej implementácií metód, ktoré budú potrebné takmer pri každej operácie vykonávanej pri tvorbe formátu OOXML.

Ďalšími triedami v návrhu sú triedy s príponou `Template`, ktoré reprezentujú šablóny pre objekty v prezentácií. Tieto triedy obsahujú XML súbory vo forme unicode stringov. V triedach typu Template sa nachádzajú šablóny pre každý XML potrebný pri tvorbe prezentácií. Takto zbytočne pri každej prezentácií nevytvárame celú štruktúru XML súborov, ale len vkladáme potrebné objekty do šablóny. Taktiež takto zaistíme isté predvolené nastavenia prezentácií, ktoré užívateľ nemusí pri každej tvorbe prezentácie zadávať.

Najdôležitejšou častou návrhu zobrazeného v diagrame tried sú konkrétne objekty, ktoré sa v prezentácií využívajú. Trieda `graph` využíva svoju šablónu pre grafy. Pomocou šablóny grafu získava trieda XML súbory, do ktorých je potrebné vložiť data na základe vstupu. Objekty vytvorené ako inštancie tejto triedy budú určené na vkladanie do objektov typu `slide`, ako je viditeľné i na obrázku. Ďalšími objektmi určenými na vkladanie do objektov typu `slide` sú objekty vytvorené ako inštancie tried `table`, `image`, `textArea` atď. Tieto triedy však už nevyužívajú šablóny, lebo neprestavujú samostatné XML súbory. Jedná sa vždy o sériu elementov vložených do XML súboru. Dôležitou triedou je trieda `relationship`, ktorá prestavuje vzťahy medzi časťami balíčka OOXML. Trieda `presentation` zaobaľuje celú prezentáciu. Táto trieda má k dispozícií šablóny súborov pre štatistiky, nastavenia, vzťahy a hlavný XML súbor prezentácie. Hlavnou funkciou však zostáva pridávanie objektov typu `slide` do prezentácie.



Obr. 4.1: Diagram tried

4.3 Ciele

V tejto časti budú stanovené ciele, ktoré chceme dosiahnuť. Pri implementácií nástroja sa zameriame na nasledujúce činnosti:

- Vytvorenie **základnej štruktúry** prezentácie, ktorá sa skladá z nasledovných častí:
  - Súbor `.rels`, uložený v zložke `_rels`.
  - Štatistické súbory `app.xml` a `core.xml` uložené v zložke `docProps`.
  - Vytvorenie štruktúry zložky `ppt`, ktorá reprezentuje obsah prezentácie - súbory slide, slideMaster, slideLayout, theme, charts ...
  - Súbor `[Content_types].xml`.
- Vytvorenie **slidov**:
  - Vytvorenie samotného XML súboru pre slide.
  - Naviazanie na zadaný slideLayout.
  - Vkladanie objektov do slidu.
- Vytváranie **objektov pre slidy**:
  - Textové polia (prispôsobiteľná veľkosť, pozícia).
  - Texty pre textové polia (prispôsobiteľné vlastnosti - tučné, kurzíva, podčiarknuté, použitý jazyk, s/bez odrážky ...).
  - Špeciálne texty - nadpisy, podnadpisy, číslovanie slidov.
  - Obrázky (prispôsobiteľná veľkosť, pozícia) - zmenšenie/zväčšenie obrázku.
  - Grafy (3 typy grafov - stĺpcový, koláčový, spojnicový).
  - Tabuľky.
  - Pridávanie vzťahov na základe pridaných objektov.
- Vytvorenie celej **prezentácie**:
  - Zvolenie grafickej témy prezentácie (dva typy grafických tém).
  - Vkladanie slidov do prezentácie.
  - Tvorba vzťahov medzi časťami prezentácie.
  - Zachovanie konzistencie - za každých okolností musí byť súbor validný.

Kapitola 5

Implementácia

V tejto kapitole sa budeme zaoberať implementáciou nástroja pre automatickú tvorbu prezentácií. Budú tu popísané použité prostriedky pri vývoji, detaily implementácie a vstupné rozhranie vytvoreného nástroja.

5.1 Implementačné prostriedky

Nástroj je implementovaný v programovacom jazyku Python (verzia 2.7) [13]. Python je dynamický objektovo orientovaný skriptovací programovací jazyk. Tento jazyk poskytuje objektovo orientovaný prístup a taktiež množstvo nástrojov na spracovanie XML súborov. Pri parsovaní a vytváraní XML súborov je využívaný modul libxml2 [4], ktorý má objektovo orientovanú reprezentáciu XML dokumentov (DOM) a podporuje jazyk XPath [18]. Pri úprave (zväčšovaní/zmenšovaní) obrázkov vložených do prezentácie je využívaný modul Image, ktorý je súčasťou Python Imaging Library (PIL) [5]. Celý nástroj bol vytváraný v prostredí Eclipse [2] za použitia pluginu PyDev [3].

5.2 Popis implementácie

Implementácia nástoja pozostáva z modulov v programovacom jazyku Python. Každý modul obsahuje jednu triedu, ktorá reprezentuje objekt použitý v prezentácii. Implementácia vychádza z návrhu z predchádzajúcej kapitoly. V tejto časti bude popísaný konkrétny postup pri implementácií jednotlivých problémov pri tvorbe prezentácií. Text bude prechádzať od obecnejších problémov ku konkrétnym.

Vytvorenie šablón pre XML dokumenty

V knižnici celú šablónu reprezentujú tri triedy, a to `slideTemplate`, `graphTemplate` a `presentationTemplate`. Šablóna pre slide obsahuje dva základné členy - šablónu pre XML súbor slidu (napr. slide1.xml) a šablónu pre súbor uchovávajúci vzťahy slidu (napr. slide1.xml.rels). Šablóna pre grafy obsahuje šablónu pre XML súbor grafu (napr. chart1.xml), šablónu pre súbor uchovávajúci vzťahy grafu (napr. chart1.xml.rels) a elementy, ktoré reprezentujú tri typy podporovaných grafov. V poslednej šablóne sa nachádzajú šablóny pre ostané súbory prezentácie. Najvýznamnejšími sú šablóny pre súbory `[Content_types].xml` a `presentation.xml`. Ďalej sa tu nachádzajú dva predvolené typy predlôh slidov (slideLayouts), predlohy podkladov (slideMasters), dve grafické témy, nastavenia vlastností prezen-

tácie (súbory `presProps.xml`, `viewProps.xml` a `tableStyles.xml`), štatistické a relačné súbory. Šablóny nám uľahčia prácu pri vytváraní XML súborov.

Reprezentácia XML súborov

XML súbor reprezentuje trieda s názvom `libxml2Document`, ktorá ukladá objektovú reprezentáciu dokumentu, vytvorenú pomocou modulu `libxml2` a metódy `parseDoc(string)`. Pre využitie dotazovacieho jazyka XPath je však potrebné vytvoriť kontext XML dokumentu a zaregistrovať menné priestory, ktoré sa v danom dokumente používajú. Kontext dokumentu vytvoríme volaním metódy `xpathNewContext` nad objektom reprezentujúcim rozparsované XML. XML dokument je potom jednoznačne určený objektovou reprezentáciou XML a kontextom so zaregistrovanými priestormi mien.

Vytváranie slidov

XML súbor pre slide vytvoríme za pomoci pripravenej šablóny. Šablóna reprezentuje prázdny slide bez objektov. Samotný XML súbor však nie je jedinou časťou vytvárania slidu v prezentácií. Potrebné je vytvoriť súbor, ktorý ukladá vzťahy slidu. Do tohto súboru sa vloží referencia na predlohu slidu. Pre úplné začlenenie slidu do prezentácie je potrebné celý balíček oboznámiť o existencií slidu (pridať referenciu na slide do prezentácie). Táto činnosť už však nepatrí do vytvárania slidu. V implementácií teda objekt slide predstavuje len XML súbor pre slide a XML súbor pre vzťahy slidu s inými časťami prezentácie. Názvy XML súborov sa určia inkrementálnym spôsobom podľa už vytvorených slidov - slide1.xml a slide1.xml.rels, slide2.xml a slide2.xml.rels atď.).

Vytváranie objektov pre slidy

Objekt, ktorý chceme do slidu vložiť sa vkladá do elementu `p:spTree`, ktorý prestavuje strom objektov na danom slide. Tento element je definovaný v šablóne slidu. Pri pridávaní objektu do slidu potrebujeme získať referenciu na element `p:spTree`. Referenciu je najľahšie získať jednoduchým XPath dotazom. XPath dotaz môžeme volať nad kontextom XML dokumentu za pomoci metódy `xpathEval(query)`. Po získaní referencie môžeme do stromu pridať nový objekt. Podľa názvu elementu, ktorý vložíme do `p:spTree`, určíme o aký objekt sa bude jednať. Po analýze objektov však vieme, že objekty vkladané do slidu majú množstvo spoločných vlastností. Jednou z najdôležitejších spoločných vlastností je element `p:cNvPr`, ktorý obsahuje atribúty `id` a `name`. Už z názvov atribútov vyplýva, že sa jedná o identifikáciu a názov objektu v rámci slidu. Tejto spoločnej vlastnosti sa dá následne využiť pri prideľovaní identifikátoru a mena novému objektu. Pomocou XPath potom zistíme všetky identifikátory a mená objektov na slide, na základe ktorých pridelíme nový unikátny identifikátor a meno objektu v rámci jedného slidu. Ďalšou spoločnou vlastnosťou je nastavovanie typu a indexu objektu v slide. Typom objektu môžeme definovať špeciálne objekty, ako nadpisy, podnadpisy atď.(sú definované v predlohe slidu - slide sa mapuje na predlohu). Pojem index je ďalšou možnosťou mapovania slidu na predlohu - predloha obsahuje niekoľko indexov, do ktorých sa objekty môžu mapovať. Zobrazenie objektu a predvolené vlastnosti pre textové pole závisí práve od typu textového poľa a jeho indexu. V implementácií je využité mapovanie na nadpisy, podnadpisy a dva typy indexov. Ďalšou spoločnou vlastnosťou je element s názvom `p:spPr`. Do elementu sa definujú viditeľné vlastnosti objektu na slide. Pomocou elementu `a:xfrm` je možné definovať rozmery a umiestnenie objektu v rámci slidu. V nástroji je implementované vytváranie nasledujúcich objektov:

- **Textové polia** - ich koreňovým elementom je element `p:sp`. Pri elemente je potrebné nastaviť neviditeľné vlastnosti objektu (element `p:nvSpPr`), kde sa definuje identifikátor a meno objektu, typ objektu a jeho index atď. Voliteľnou záležitosťou je zadanie rozmerov a umiestnenia textového poľa. Pri nezadaní týchto údajov sa uplatňujú predvolené vlastnosti. Pre pridanie textu do textového poľa je potrebné vytvoriť podlement `txBody` (Text Body), do ktorého vložíme element `a:p`, ktorý reprezentuje odstavec v texte. Celý odstavec textu sa potom skladá z jednotlivých slov, ktorým môžeme definovať vlastnosti. V implementácií pre vkladanie textu rozdeľujeme text do odstavcov a tie do jednotlivých slov (element `a:r`(Text Run)), ktoré majú svoje vlastnosti. Týmto dosiahneme možnosť formátovania jednotlivých slov v rámci odstavca.

- **Obrázky** - ich koreňovým elementom je element `p:pic`. Element, ktorý definuje neviditeľné vlastnosti pre obrázok sa nazýva `p:nvPicPr`. Na rozdiel od textového tela v textovom poli, pri obrázkoch vkladáme referenciu na obrázok. Referenciu predstavuje identifikátor vzťahu slidu s konkrétnym obrázkom. Referenciu vytvoríme pridaním elementu `p:blipFill` (Picture Fill) a jeho podelementu `a:blip`, ktorý obsahuje atribút `r:embed`. Hodnotu atribútu bude obsahovať identifikátor vzťahu. Pri vložení obrázku do slidu z pravidla volíme veľkosť a umiestnenie obrázku. Tu nastáva problém s upravením veľkosti obrázku. Program si obrázok musí skopírovať do dočasného priečinku, kde obrázok upraví na požadovanú veľkosť.

- **Grafické rámce** - pri vytváraní tabuliek a grafov je potrebné objekty vkladať do grafických rámcov (elementu `p:graphicFrame`). Vytvorenie grafického rámca je takmer zhodné s vytvorením rámca pre obrázok alebo text. Určujeme mu teda identifikáciu, mapovanie, umiestnenie a rozmery. Konkrétny objekt budeme vkladať do elementu `a:grafic`. Údaje pre objekt sa vkladajú do elementu `a:graficData`, ktorý obsahuje atribút `uri`, pomocou ktorého definujeme typ objektu v grafickom rámci.

- **Tabuľky** - tabuľka sa v slide vkladá do grafického rámca. Do atribútu `uri` elementu `a:graficData` vložíme XML schému pre tabuľku - „http://schemas.openxmlformats.org/drawingml/2006/table“. Koreňovým elementom tabuľky je `a:tbl`. Na začiatku sa definujú vlastnosti tabuľky pomocou elementu `a:tblPr`. V elemente `a:tblGrid` definujeme šírku buniek tabuľky - vypočíta sa z šírky grafického rámca. Po nastaveniach tabuľky je potrebné vložiť konkrétne údaje pomocou elementov `a:tr` (Table Row) a `a:tc` (Table Cell), pomocou ktorých definujeme riadky a konkrétne bunky v tabuľke. Do buniek sa vkladajú údaje - konkrétny text.

- **Grafy** - graf, podobne ako tabuľka, sa pridáva do grafického rámca. Rozdielne údaje definuje atribút `uri` elementu `a:graficData`, v ktorom je XML schému pre graf „http://schemas.openxmlformats.org/drawingml/2006/chart“. Na rozdiel od tabuliek, grafy majú údaje uložené v samostatnom XML súbore. Do slidu teda pridáme len referenciu pomocou identifikótoru vzťahu, ktorý predstavuje spojenie slidu a údajov pre graf. Táto referencia sa vkladá do elementu `c:chart`. Ďalšou úlohou pri vytváraní grafu je vytvorenie XML súboru, ktorý by ukladal údaje pre graf. Jedná sa o súbor `chartXY.xml`, kde XY je poradové číslo grafu v prezentácií. Vytváranie tohto XML súboru je v nástoji implementované za pomoci šablony, ktorá sa nastaví podľa typu grafu, ktorý chceme použiť. Po vytvorení základu XML vkladáme do grafu údaje. Údaje sa vyberajú s XLSX súboru, ktorý zadáva uživateľ. Program XLSX súbor rozbalí do dočasného priečinku a získa zo zadaných buniek údaje (napr. užívateľ zadá

datovú oblasť Sheet1!$A$2:$A$5 - program teda vyhľadáva v XML súbore `/xl/worksheets/sheet1.xml`) údaje z buniek A2 až A5). V XML súbore pre reprezentáciu údajov pre graf vytvárame série údajov (element `c:ser`) a konkrétne údaje ukladáme do podelementu `c:val`. Taktiež je možné vkladať popisky grafov za pomoci elementu `c:tx` - popis sérií údajov a elementu `c:cat` - popis kategórií údajov. Pre XML súbor reprezentujúci údaje pre graf je potrebné vytvoriť spojenie s XLSX súborom obsahujúcim údaje - pridá sa vzťah medzi súbormi. XLSX súbor sa teda stáva súčasťou prezentácie (nakopíruje sa do zložky `/ppt/embeddings/`). Potom pri editácií údajov grafu napríklad z Microsoft PowerPoint sa XLSX súbor otvorí a je možné ho editovať.

Vytvorenie celej prezentácie

Doposiaľ tu boli popísané postupy ako program vytvára slidy a objekty vyskytujúce sa na slidoch. Konečné vytvorenie prezentácie predstavuje tri činnosti:

- Vloženie slidov do prezentácie - uloženie referencií na slide do súborov `presentation.xml` a `[Content_types].xml`
- Doplnenie štatistických súborov a súborov ukladajúcich nastavenia prezentácie.
- Vytvorenie štruktúry zložiek, nakopírovanie všetkých potrebných súborov (obrázky, XLSX súbory ...), vytvorenie XML súbor a zabalenie do ZIP archívu.

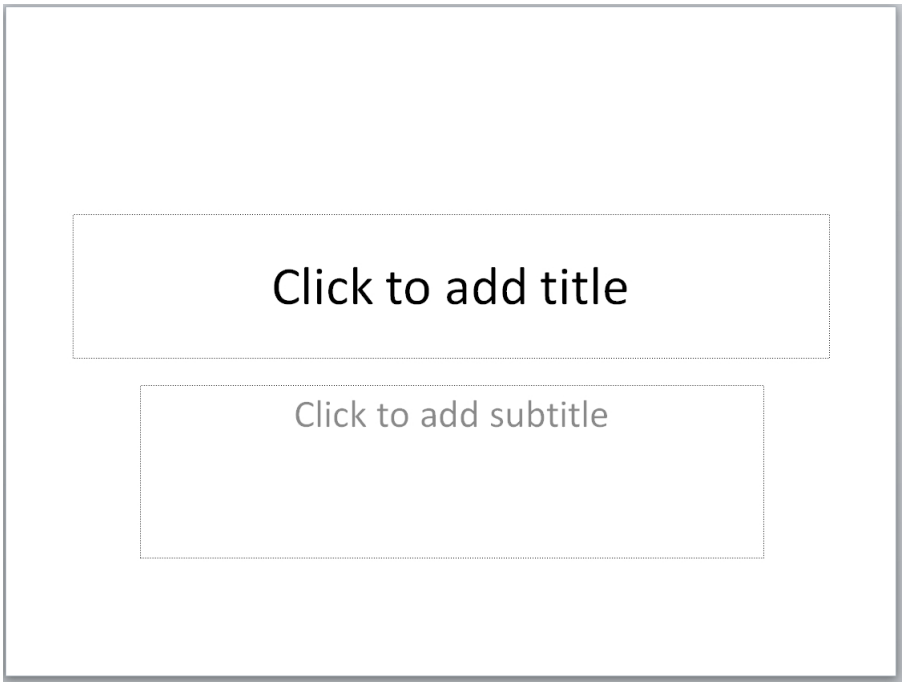
5.3 Vstupné rozhranie

V tejto časti bude popísané vstupné rozhranie nástroja. Bude tu popísaný spôsob tvorby prezentácií vo vytvorenom nástroji a spôsob tvorby objektov v prezentácii.

Pri tvorbe prezentácie je potrebné importovať dve základné triedy. Triedu **presentation** z modulu `presentation` a triedu **slide** z modulu `slide`. Týmto sa užívateľovi knižnice poskytuje možnosť vytvárať objekty typu slide, a teda slidy určené pre prezentáciu a objekty typu presentation, ktoré reprezentujú celú prezentáciu.

Pri vytváraní objektov typu slide je potrebné zavolať konštruktor triedy. Konštruktor má dva voliteľné parametere. Parameter `slideLayout` - do parametru môžeme špecifikovať typ rozloženia slidu. Prípustné sú dve hodnoty - „Title Slide“ a „Title and Content“. Rozloženie slidov je demonštrované na Obr. 5.1 a Obr. 5.2. Pri opomenutí parametru `slideLayout` sa použije predloha s názvom „Title and Content“. Ďalším parametrom je `slideNumber`, kde je možné explicitne zapnúť alebo vypnúť zobrazovanie poradového čísla slidu (prístupné hodnoty sú 1 a 0). Po vytvorení objektu typu slide má užívateľ dostupné nasledujúce metódy:

- **createTextArea()** - vloží do slidu textové pole. Návratovou hodnotou je objekt, ktorý textové pole reprezentuje. Táto hodnota je neskôr potrebná pri vkladaní textu do textového poľa.
- **createParagraph(textArea, level)** - vytvorí odstavec v textovom poli. Parameter `textArea` je návratová hodnota volania metódy `createTextArea()`. Voliteľným parametrom je parameter `level`, ktorý určuje odsadenie odstavcu (odrážku) - stupeň 0 je odrážka s najväčšou prioritou, stupeň 1 s menšou atď.



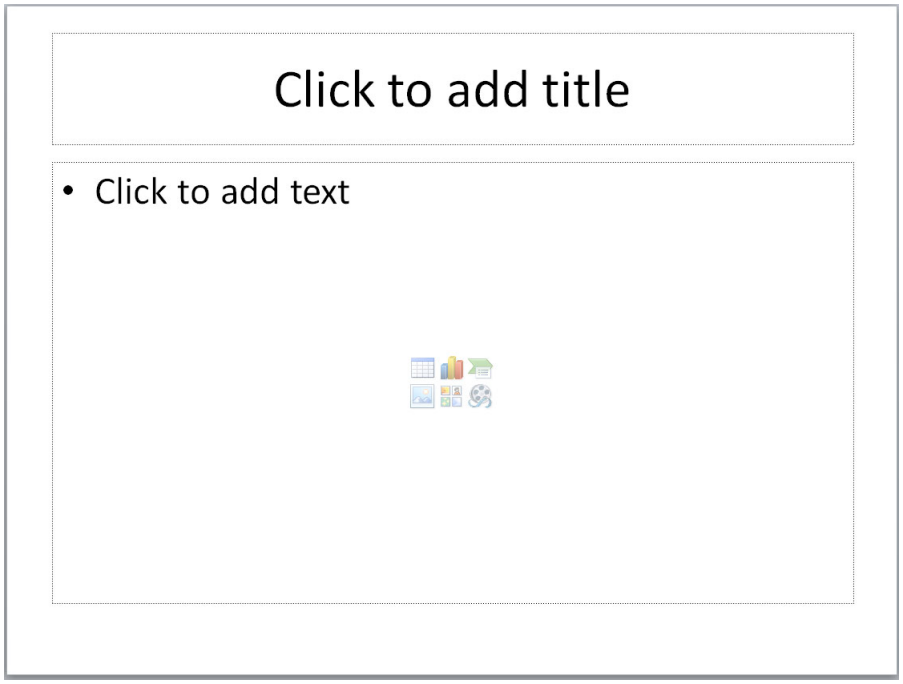
Obr. 5.1: Predloha slidu s názvom „Title Slide“

- **addText(paragraph, content, properties)** - pri volaní metódy je užívateľ povinný zadať parameter `paragraph` - objekt vytvorený volaním `createParagraph()` (pre vloženie textu je potrebný odstavec, do ktorého sa text vloží) a parameter content - konkrétny text, ktorý do textového poľa chceme vložiť.

  Nepovinný parameter `properties` určuje vlastnosti textu, ktorý vkladáme. Vlastnosti vkladáme formou Python Dictionary (slovník) [1]. Príklad zadania - properties = {"vlastnosť": "hodnota"}. Vlastnosti, ktoré text môže nadobúdať je veľké množstvo a sú popísané v odkaze [11]. Vlastnosti textu, ktoré sa využívajú najčastejšie:

  - b (Bold) - tučné písmo, hodnotou vlastnosti je buď 0 alebo 1.
  - i (Italics) - kuzíva, hodnotou vlastnosti je buď 0 alebo 1.
  - u (Underline) - podčiarknutie textu, prípustných hodnôt je mnoho typov -„sng“ (rovná plná čiara), „wavy“ (vlnovka), „dash“ (prerušovaná čiara) atď. Všetky typy hodnôt je možné nájsť v zdroji [11].
  - sz (Font Size) - veľkosť písma.
  - lang (Language ID) - špecifikácia jazyka textu (napr. en-US, cs-CZ).
  - noProof (No Proofing) - text nemá byť kontrolovaný na gramatické chyby.
  - font - definovanie použitého fontu. Vlastnosť sa definuje do elementu `a:latin`.

- **addTitle(content, properties, type)** - metóda vkladá nadpis slidu. Jediným povinným parametrom je `content` - samotný text nadpisu. Parametre `properties` a `type` sú voliteľné. Keďže sa jedná o text, parameter `properties` určuje vlastnosti textu rovnako pri vkladní textu do textových polí. Parameter `type` určuje typ nadpisu - title (hlavný nadpis), subtitle (podnadpis). Pri nezadanom parametri sa použije predvolená hodnota title.



Obr. 5.2: Predloha slidu s názvom „Title and Content“

- **addImage(imagePath, width, height, locX, locY)** - metóda vkladá do slidu obrázok. Povinným parametrom je `imagePath`, ktorý jednoznačne určuje umiestnenie a názov obrázku vkladaného do slidu. Voliteľné parametre sa týkajú veľkosti a umiestnenia obrázku (hodnoty sa zadávajú v pixeloch):
  - `width` - šírka obrázku.
  - `height` - výška obrázku.
  - `locX` - umiestnenie ľavého horného rohu obrázku na pomyselnej osi X slidu.
  - `locY` - umiestnenie ľavého horného rohu obrázku na pomyselnej osi Y.
- **addGraph(type, xlsxFile, data, width, height, locX, locY):** - metóda vkladá do slidu graf. Všetky parametre metódy sú povinné. Popis parametrov:
  - `type` - typ grafu, prípustné hodnoty sú barChart (stĺpcový graf), pieChart (koláčový graf) a lineChart(spojnicový graf).
  - `xlsxFile` - cesta a názov súboru XLSX, z ktorého budú použité údaje pre graf.
  - `data` - zadáva sa formou Python Dictionary (slovník), kde kľúč je názov série údajov a hodnota je rozsah buniek z XLSX súboru. Príklad zadania údajov - dataForGraph = {”data1”: ”Sheet1!$B$2:$B$5”}.
  - `width` - šírka grafického rámca, do ktorého sa graf vykreslí.
  - `height` - výška grafického rámca, do ktorého sa graf vykreslí.
  - `locX` - umiestnenie ľavého horného rohu grafického rámca na pomyselnej osi X slidu.

  - `locY` - umiestnenie ľavého horného rohu grafického rámca na pomyselnej osi Y slidu.

- **addTable(data, countRow, countColumn, width, height, locX, locY)** - metóda vkladá do slidu tabuľku. Všetky parametre metódy sú povinné. Popis parametrov:

  - `data` - dvojrozmerné pole, ktoré reprezentuje tabuľku.
  - `countRow` - počet riadkov tabuľky.
  - `countColumn` - počet stĺpcov tabuľky.
  - `width` - šírka grafického rámca, do ktorého sa tabuľka vykreslí. Bunky v tabuľke budú mať potom šírku rovnú šírka_bunky = width/countColumn. Každá bunka bude mať rovnakú šírku.
  - `height` - výška grafického rámca, do ktorého sa tabuľka vykreslí. Bunky v tabuľke budú mať potom výšku rovnú výška_bunky = height/countRow. Každá bunka bude mať rovnakú výšku.
  - `locX` - umiestnenie ľavého horného rohu grafického rámca na pomyselnej osi X slidu.
  - `locY` - umiestnenie ľavého horného rohu grafického rámca na pomyselnej osi Y slidu.
  - `data` - dvojrozmerné pole, ktoré reprezentuje tabuľku.

K vytvoreniu objektu **presentation** je potrebné zavolať konštruktor triedy **presentation**. Konštruktor má dva nepovinné parametre. Prvým je parameter `theme` - určuje použitie grafickej témy v slide (prípustné hodnoty sú „Adjacency“ a „Flow“). Grafické témy sú prebraté z nástroja Microsoft PowerPoint (v iných programoch ich zobrazenie nemusí fungovať správne). Pri opomenutí tohto parametra prezentácia nebude obsahovať žiadnu grafickú tému. Druhým parametrom je `slideNumber` - určuje použitie číslovania slidov v prezentácii. Prípustnými hodnotami sú 1 a 0, ktoré signalizujú vloženie resp. opomenutie číslovania slidov. Ak je vloženie číslovania slidov zapnuté, uplatňuje sa na celú prezentáciu, okrem slidov kde je to explicitne nastavené inak. Pri nezadaní tohto parametru sa číslovanie slidov nevkladá. Po vytvorení objektu typu **presentation** je možné nad objektom volať nasledujúce metódy:

- **addSlide(slide)** - vloží do prezentácie slide. Jediným povinným parametrom je objekt odvodený od triedy slide.

- **Print(outputPath)** - vytvorí súbor prezentácie. Jediným povinným parametrom je cesta a názov súboru, ktorý sa má vytvoriť.

Kapitola 6

Výsledky

V tejto kapitole budú zhodnotené výsledky práce. Výsledkom je nástroj pre automatickú tvorbu prezentácií, vytvorený v programovacom jazyku Python. V nasledujúcom texte budú ukázané experimenty s nástrojom, demonštrované možnosti použitia a stručná ukážka použitia.

6.1 Vytvorenie prezentácie

Prezentáciu vytvoríme sekvenciou príkazov v programovacom jazyku Python. V Tabuľka 6.1 je znázornený import potrebných tried. Potrebné sú dve základné triedy - `slide` a `presentation`. Po importovaní tried môžme vytvoriť slide. Spôsob vytváranie slidu je naznačený v Tabuľka 6.2. V tabuľke je naznačené vytvorenie slidu, vloženie nadpisu, vytvorenie textové poľa a odstavcov v textovom poli, vloženie konkrétnych textov do odstavcov a vloženie obrázku. Takto môžeme vytvoriť slide, ktorý je zobrazený na Obr. 6.1. Na obrázku vidíme nadpis slidu, štyri odstavce s rôznymi úrovňami odsadenia, kde v jednotlivých slovách odstavcov sú využité rôzne rezy písma. Ďalej sa tu nachádza obrázok so zadanými rozmermi a pozíciou a číslo slidu.

Na Obr. 6.2 je zobrazený slide, ktorý obsahuje dva grafy (koláčový a stĺpcový). Ďalej sa tu nachádza tabuľka s rozmermi 3x3. Zápis vytvorenia grafu a tabuľky, následné vytvorenie celej prezentácie pridanie slidov do prezentácie a konečné vytvorenie výstupu aplikácie je demonštrované v Tabuľka 6.3. Údaje pre graf reprezentuje premenná `dataForGraph`, ktorá je slovníkom s kľúčmi (názvy údajov) a hodnotami (oblasti buniek). Údaje pre tabuľku reprezentuje premenná `dataForTable`, ktorá je dvojrozmerným poľom (v tomto prípade matica 3x3). Prezentácia bude vytvorená s grafickou témou „Flow“ a bude obsahovať dva slidy.

```
sys.path.append("./modules")
from presentation import presentation
from slide import slide
```

Tabuľka 6.1: Import potrebných tried

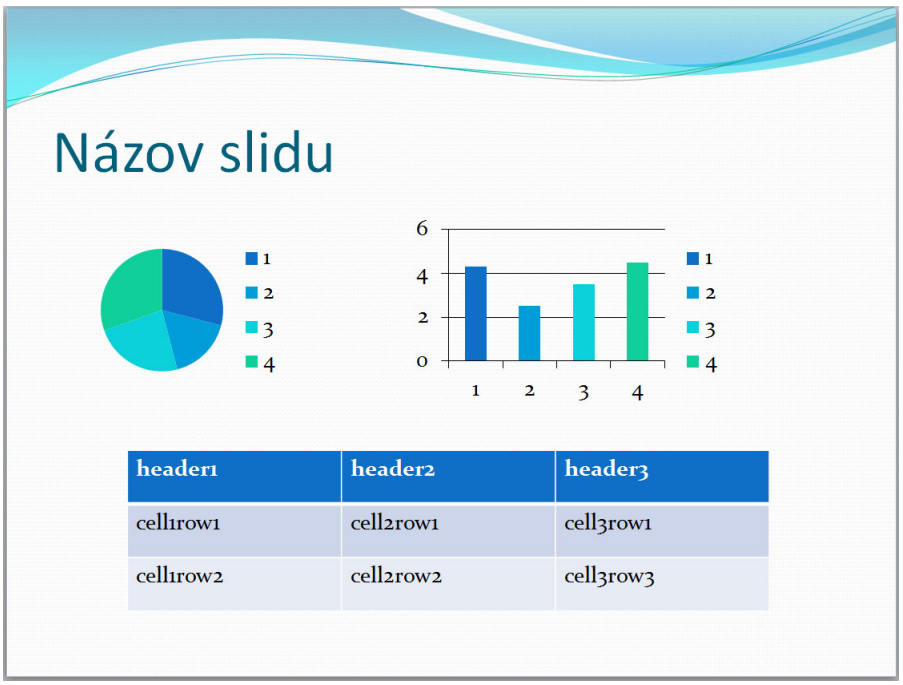
```
...
slide1 = slide()
slide1.addTitle(content=”Nadpis slidu”)
textArea = slide1.createTextArea()
paragraph1 = slide1.createParagraph(textArea, level=0)
paragraph2 = slide1.createParagraph(textArea, level=1)
...
slide1.addText(paragraph1, content=”Kurziva”, properties = ”i”:”1”)
slide1.addText(paragraph1, content=”podciarknute”,
               properties = {”u”:”sng”, ”font”": ”Centaur”})
...
slide1.addImage(imagePath="Hydrangeas.jpg",
                width=400,
                height=300,
                locX=300,
                locY=500)
```

Tabuľka 6.2: Vytvorenie slidu a demonštrácia práce s textom a obrázkom v slide

```
...
slide2.addGraph(type=”pieChart”,
                xlsxFile=”ExcelFile.xlsx”,
                data=dataForGraph,
                width=800,
                height=500,
                locX=100,
                locY=230)
...
slide2.addTable(data=dataForTable,
                countRow=3,
                countColumn=3,
                width=800,
                height=200,
                locX=150,
                locY=550)
...
presentation1 = presentation(theme=”Flow”)
presentation1.addSlide(slide1)
presentation1.addSlide(slide2)
presentation1.Print(”myPresentation.pptx”)
```

Tabuľka 6.3: Vloženie grafu, tabuľky a vytvorenie celej prezentácie

Obr. 6.1: Príklad slidu s textom a obrázkom (grafická téma „Flow“)



Obr. 6.2: Príklad slidu s grafmi a tabuľkou (grafická téma „Flow“)

Kapitola 7

Záver

Výsledkom práce je implementácia nástoja pre automatickú tvorbu prezentácií uložených vo formáte OOXML.

Okrem vlastnej implementácie práca zahŕňa teoretický rozbor danej problematiky. Popísaný je jazyk XML, na ktorom je formát OOXML založený, formát OOXML a jeho vlastnosti a podrobne časť formátu zameraná na ukladanie prezentačných súborov.

Cieľom práce bolo vybrať vhodnú podmnožinu funkcií prezentácií a vytvoriť nástroj prenositeľný medzi rôznymi platformami použiteľný pri tvorbe prezentácií.

Prínos práce predstavuje zoznámenie sa s formátom OOXML a jeho časti ukladajúcej prezentácie a moduly v programovacom jazyku Python využiteľné na tvorbu jednoduchých prezentácií pri automatickej tvorbe prezentácií. Nástroj je určený pre aplikácie, ktoré by automaticky generovali prezentácie zo vstupných údajov. Možné využitie nástoja je pre ľudí, ktorí denne potrebujú vytvárať prezentácie podobného formátu z rôznych výsledkov svojej práce. Demonštrácia použitia nástroja sa nachádza v Kapitole 6.

7.1 Návrh pokračovania projektu

Nástroj vytvorený v práci podporuje len najpoužívanejšie elementy prezentácií. Pri špecializovaní sa na určité typu prezentácií (množstvo rôznych grafov, animácie atď.) by bolo potrebné funkcie nástroja rozšíriť. Pre využitie nástroja v praxi by bolo vhodné nad vytvoreným nástrojom vytvoriť aplikáciu, ktorá by analyzovala údaje a na základe údajov automaticky vytvorila prezentácie. Takáto aplikácia by uľahčila tvorbu prezentácií ľudom, ktorý prezentácie vytvárajú denne z výsledkov svojej práce. Pre vytváranie bežných prezentácií je vhodnejšie využiť nástroje s grafickým užívateľským rozhraním (Microsoft PowerPoint atď.).

Literatúra

[1] Built-in Types. http://docs.python.org/library/stdtypes.html, [cit. 2012-04-05], [online].

[2] Eclipse - The Eclipse Foundation open source community website. http://www.eclipse.org/, [cit. 2012-04-23], [online].

[3] PyDev. http://pydev.org/, [cit. 2012-04-02], [online].

[4] Python and bindings. http://xmlsoft.org/python.html, [cit. 2012-04-03], [online].

[5] Python Imaging Library (PIL). http://www.pythonware.com/products/pil/, [cit. 2012-04-07], [online].

[6] SAX. http://www.saxproject.org/, [cit. 2012-03-28], [online].

[7] Bradley, N.: *XML pro každého*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1996, ISBN 0-201-41999-8.

[8] ECMA: Standard ECMA-376 Office Open XML File Formats. http://www.ecma-international.org/publications/standards/Ecma-376.htm, [cit. 2012-03-02], [online].

[9] Harold, E. R.: *XML: Extensible Markup Language*. IDG Books Worldwide, Inc. Foster City, CA, 2000, ISBN 07-6453-199-9.

[10] ISO/IEC: Information Technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane (ISO/IEC 10646:1993). 1993.

[11] ISO/IEC: *29500 Part 1: Fundamentals and Markup Language Reference*. Second edition vydání, 2011, ISBN 80-85615-77-0, [online pdf].

[12] Kosek, J.: *XML pro každého*. Grada Publishing, 2000, ISBN 80-7169-860-1.

[13] Lutz, M.: *Programming Python*. O'Reilly Media, 2006, ISBN 05-9600-925-9.

[14] Microsoft: Výuka k systému Office. http://office.microsoft.com/cs-cz/training/, [cit. 2012-03-07], [online].

[15] Microsoft: Working with PresentationML Documents. http://msdn.microsoft.com/en-us/library/gg278318.aspx, [cit. 2012-03-10], [online].

[16] (W3C), W. W. W. C.: Document Object Model (DOM). http://www.w3.org/DOM/, [cit. 2012-03-28], [online].

[17] (W3C), W. W. W. C.: Extensible Markup Language (XML). http://www.w3.org/XML/, [cit. 2012-03-15], [online].

[18] (W3C), W. W. W. C.: XML Path Language (XPath). http://www.w3.org/TR/xpath/, [cit. 2012-03-28], [online].

Dodatok A

Obsah CD

\src	zdrojové súbory aplikácie
\demo	demoaplikácie vytvorené pomocou implementovaného nástroja
\virtual_machine	virtuálny stroj (Virtual Box - Ubuntu 10.10) s demoaplikáciami na ploche (predinstalované potrebné knižnice)
\doc	zdrojové súbory technickej správy v LaTeX
\doxygen	dokumentácia zdrojových kódov
manual.pdf	stručný manuál na použitie aplikácie
BP.pdf	technická správa
readme.txt	nápoveda k obsahu CD